TP 2 - Programmation mobile avancée

SANCHEZ Martin

Exercice 1 - Providers et Bloc / Cubit

Pour cet exercice nous avons repris l'exercice précédent. Il s'agissait de reproduire le même logique métier avec différents concepts, à savoir les providers et les blocs.

Pour rappel, il s'agissait d'une application de quizz, l'interface se présente comme ceci :

FIGURE 1 – Interface de l'application de quizz

Cette interface affiche selon les question une image, la question en elle même, et trois boutons, ces boutons permettent de répondre à la question.

Une fois la question répondu, les boutons s'affichent en vert pour la bonne réponse, rouge pour la mauvaise.

Le dernier bouton permet de passer a la question suivante.

Providers

Pour cet exercice nous avons ajoute une classe DataProvider qui s'occupe de gérer les données et de notifier les clients des changements.

Cette classe étends ChangeNotifier, et implémente toutes les valeurs qui vont être amenés a être modifiés dans la page. C'est a dire la question, la couleur des boutons. il contient aussi les méthodes pour modifier ces valeurs et notifier les clients.

On s'abonne au provider dans notre interface avec : "Provider.of<QuizzModel>(context, listen : true)". Le booléen "listen" permet de dire si c'est une donnée, ou une méthode.

Bloc / Cubits

Pour cette deuxième façon de faire, nous utiliserons les bloc. Ces deux méthodes sont assez similaires. Pour les bloc, nous allons créer une classe qui représente un état. Avec a l'intérieur tout les éléments qui changent.

On créer ensuite une autre classe qui hérite d'un Cubit de notre état (Classe paramétrée). C'est ce cubit qui va être amené a modifier l'état et a le propager aux clients.

Nous récupérerons ces informations avec la ligne : "context.read<HomepageCubit>()"

Exercice 2 - Meteo

Dans le second exercice nous avons réalisé une application de prévision météo. Pour récupérer les données météo nous utilisons l'API de https://openweathermap.org/. Pour cela nous encapsulons dans une classe tout les accès au réseau.

Cette api nous fourni des données en JSON, et nous utilisons une factory JSON dans nos modèles de données météo pour reconstruire nos objets.

Nous avons 3 appels à l'API, un premier permet de récupérer des données GPS a partir du nom d'une ville. Celui ci nous permet de cibler précisément une ville, et de récupérer des informations ; comme le pays par exemple.

Nous avons un second appel à l'API pour récupérer les données actuelles. Pour cela nous donnons les coordonnés récupérer grâce à l'appel précédent. Nous récupérons alors un modèle météo contenant la température, le temps, la vitesse du vent, l'humidité et la pression atmosphérique.

Le dernier appel permet de récupérer les données météo pour les prochains jours, nous récupérerons le temps, la température et la vitesse du vent.

Nous disposeront toutes ces informations dans une interface, Par défaut nous affiche-

FIGURE 2 – Interface de l'application de météo

rons les informations de la ville de Montpellier, nous pouvons imaginer comme amélioration possible, le fait d'afficher par défaut la ville de l'utilisateur selon ça position.

Le champ de texte en haut de la page permet de changer la ville, il suffit d'écrire le nom (sans fautes) et de cliquer sur la loupe. Nous utilisons aussi les cubits pour rendre nos vues dynamiques.